Gremlin System Clocking Design

Table of Contents

[System Master Clock 2](#_Toc306549717)

[System Device Clock Rates 2](#_Toc306549718)

[VGA 2](#_Toc306549719)

[Audio 2](#_Toc306549720)

[ZPU 2](#_Toc306549721)

[DMA 3](#_Toc306549722)

[Copper 3](#_Toc306549723)

[Blitter 3](#_Toc306549724)

[MMC 3](#_Toc306549725)

[Keyboard and Mouse 3](#_Toc306549726)

[RAM Access Prioritization 3](#_Toc306549727)

# System Master Clock

This project was designed to run on a n XESS XuLA-200 board, which is based on the Spartan-3A FPGA, with a built-in external clock of 12Mhz available. To avoid using up pins unnecessarily, all clocks in this design are derived from that source **12Mhz** clock, which is called **FPGA\_CLK \_IN**.

All other (internal) clocks described below, are derived from this master clock, using one or more DCM’s. A DCM requires Multiply and Divide values. Eg. M14,D24 (seen below) means ((12Mhz x 14) / 24) = 7Mhz.

Most clocks used below, are derived from the VGA’s pixel-clock rate. I chose this strategy, in hopes that it would simplify design, by allowing me to use an arbitrator to control which device gets to use the RAM’s on any particular clock, simply by using a priority rank, while all devices stay in sync and have access to specific windows of time.

# System Device Clock Rates

## VGA

The value of **36Mhz (M12,D4)** was chosen as the Pixel Clock, due to the fact that the most resolutions can be generated from it, combined with the fact that the clock rate itself can be generated from the master 12Mhz clock using a Xilinx DCM. 800x600@56Hz and 640x480@87Hz are the two primary resolutions, and pixel-doubling techniques can produce variants all the way down to 160x120.

## Audio

Two clocks are required:

1. Converts a Sample into a PWM stream (PWM\_CLK)
2. Traverses a SoundSample over time, using a period as defined by the channel (CHANNEL\_CLK).

PWM\_CLK requires a rate that is substantially faster than CHANNEL\_CLK, so that even at maximum channel speed, each sample still gets to be converted to PWM multiple times, and without audible aliasing between the two clocks.

PWM\_CLK was chosen as **2.25Mhz**, as it can be derived from the VGA’s 36Mhz by dividing by 16.

CHANNEL\_CLK was chosen as **3.56Mhz (MM8,D27)** because it lands within reasonable range of the Amiga’s audio clock rates of: NTSC = 3.579545, and PAL = 3.546895. Unfortunately, no divisible value of the FPGA\_CLK\_IN comes close to either of the target frequencies, so the audio channel system will be driven from this asynchronous clock instead. A FIFO buffer will exist between the audio channels and the RAM. The FIFO will empty when the channels request a new value, and will be given a chance to refill (when necessary) on each H-Blank, as the Amiga did. Since the VGA H-Blanks are a much higher frequency, this design should allow maximum pitch to exceed that which the Amiga was limited to (which was a period of 124, minimum). This FIFO will have to offer a way to cross clock domains, or errors might be audible.

## ZPU

By doubling the VGA’s 36Mhz to **72Mhz (M12,D2)**, we are approaching the limit of the ZPU (which , I read, was about 90Mhz?) while keeping the ZPU on a clock that’s divisible with the rest of the system.

## DMA

This system can move raw blocks of RAM from one location to another, as fast as it can find available clocks. It runs at **72Mhz**.

## Copper

This is a tiny co-processor that can manipulate memory (such as video registers or palette values) when specific pixels are visited by the VGA circuitry. It runs at **72Mhz**.

## Blitter

This system can move blocks of image memory, draw lines, and perform flood-fills. It runs at **72Mhz**.

## MMC

SD is rated at up-to 25Mhz. Although I call it an MMC device, it’s actually SD. The MMC device will operate at **18Mhz**, by dividing the 72Mhz by 4.

NOTE: I could target 24Mhz by dividing the 72Mhz by 3, to get a bit more speed.

## Keyboard and Mouse

The PS/2 ports expect to clock at between 10 and 16.7kHz. The keyboard/mouse always generate the clock signal, so it will be necessary to cross clock domains. On the CPU side, this device will communicate at the full **72Mhz**, and will likely be offering a simple collection of registers/values, and an interrupt line, to the CPU.

# RAM Access Prioritization

The devices are granted access to RAM in the following priorities:

Devices, ranked according to priority (1 = highest priority for accessing RAM).

1. VGA (preassigned fetch slots)
2. Audio (preassigned fetch slots)
3. Blitter
4. Copper
5. DMA
6. Mouse
7. Keyboard
8. MMC
9. ZPU